ALGORITMOS DE FORD Y FORD-BELLMAN

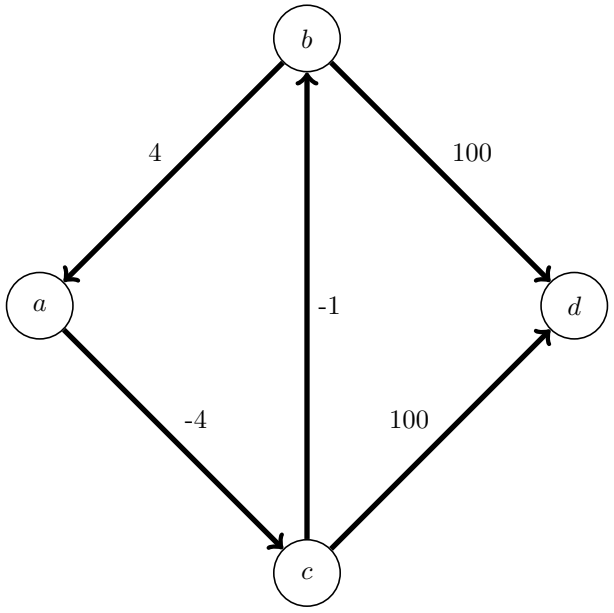
Complementos de Matemática 1

Tomás Albanesi
Milton Casas
Milagros Osimi

1 Camino más corto

El algoritmo de Ford-Bellman se utiliza para encontrar el camino más corto desde un nodo fuente a todos los demás nodos en un grafo dirigido ponderado. La principal motivación detrás de este algoritmo es su capacidad para manejar grafos que contienen aristas con pesos negativos, a diferencia del algoritmo de Dijkstra, que no funciona correctamente en tales casos.

Es importante tener en cuenta que, debido a su naturaleza de exploración exhaustiva, el algoritmo puede ser menos eficiente que otros algoritmos de camino más corto en grafos sin aristas con pesos negativos. Además el algoritmo no funciona en todos los grafos, ya que, en grafos con ciclos negativos no es posible encontrar una ruta (finita) mas corta.



Si buscamos el camino más corto en este grafo desde a hasta d no lo encontraremos, ya que, el camino $P = (a, c, b, d)$ pesa 95 pero $P' = (a, c, b, a, c, b, d)$ pesa 94 y si hacemos tender a infinito las repeticiones del ciclo $C = (a, c, b, a)$ en el camino, el mismo tiende a tener peso menos infinito. Esto nos introduce al problema del camino más corto:

1.1 El problema del camino más corto

Entrada: Un grafo dirigido $G = (V, E)$, un vértice $r \in V$, y una función de costos $w(e) : e \in E$.
Objetivo: Encontrar, para cada $v \in V$, un camino dirigido de r a v con costo mínimo, si existe.

Supongamos que tenemos un vector $y = \{y_v \in \mathbb{R} : v \in V\}$ donde y_v es un costo (cualquiera, no necesariamente mínimo) de una ruta de r a v y encontramos un arco $zu \in E$ tal que $y_z + w(zu) < y_u$. Podemos encontrar un camino de r a u usando el camino hasta z y sumando la arista zu, de costo menor a y_u. En particular, si y_v es el costo del camino más corto de r a v $\forall v \in V$ entonces se satisface que:

$$y_z + w(zu) \geq y_u \ , \forall zu \in E \ (z, u \in V)$$

Definición 1: Llamamos a $y = (y_v : v \in V)$ un potencial factible si satisface que $y_r = 0$ y

$$y_z + w(zu) \geq y_u \ , \forall zu \in E \ (z, u \in V)$$

Proposición 2: Sea y un potencial factible, y P un camino dirigido de r a v. Entonces $w(P) \geq y_v$.
Demostración: Supongamos que $P = v_0, e_1, v_1, ..., e_k, v_k$, donde $v_k = v$, $v_0 = r$. Entonces:

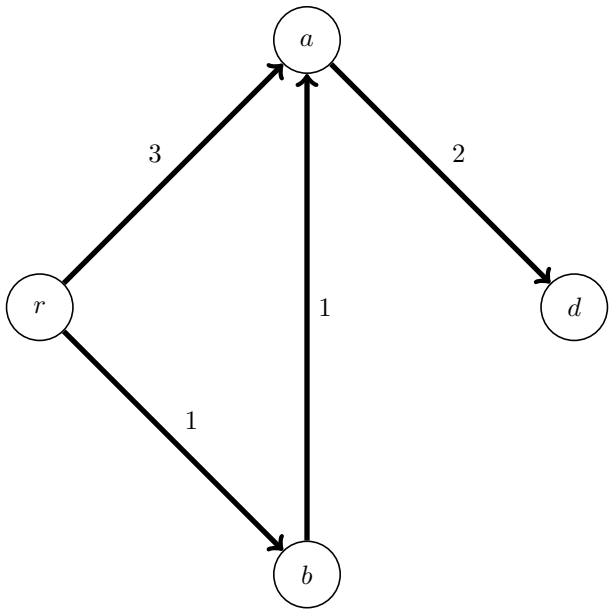
$$c(P) = \sum_{i=1}^{k} w(e_i) \geq \sum_{i=1}^{k} (y_{v_i} - y_{v_{i-1}}) = y_{v_k} - y_0 = y_v - y_r = y_v - 0 = y_v$$

Observación: Sub-caminos de caminos óptimos son caminos óptimos: Si P es un r, u-camino, se puede descomponer en dos caminos P_1 de r a v y P_2 de v a u, si P_1 no fuera un camino óptimo, podríamos intercambiar P_1 por un camino de menor costo y obtener un camino de r a u de costo menor a P.

1.2 Algoritmo de Ford

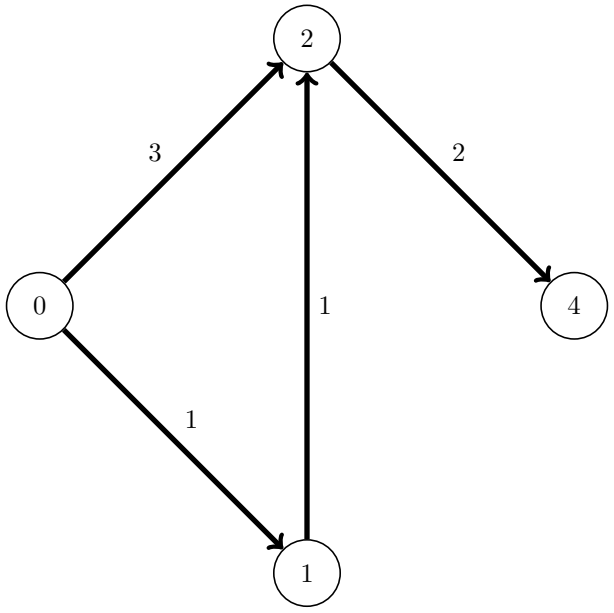
Entrada: Un grafo dirigido $G = (V, E)$, un vértice $r \in V$, y una función de costos $w(e) : e \in E$.
Inicio: Un vector y, donde $y_r = 0$, $y_v = \infty$ si $v \neq r$. Un vector p donde $p_r = 0$, $p_v = -1$ si $v \neq r$, que nos servirá para reconstruir el camino de r a v.
Iteración: Mientras y no sea un potencial factible, buscar una arista vu tal que $y_v + w(vu) < y_u$ y corregir $y_u = y_v + w(vu)$.
Salida: Un potencial factible y, donde y_v es el costo del camino más corto de r a v, y un vector p donde p_v es el predecesor de v en el camino menor costo de r a v.

Veamos algunos ejemplos:
Ejemplo 1:

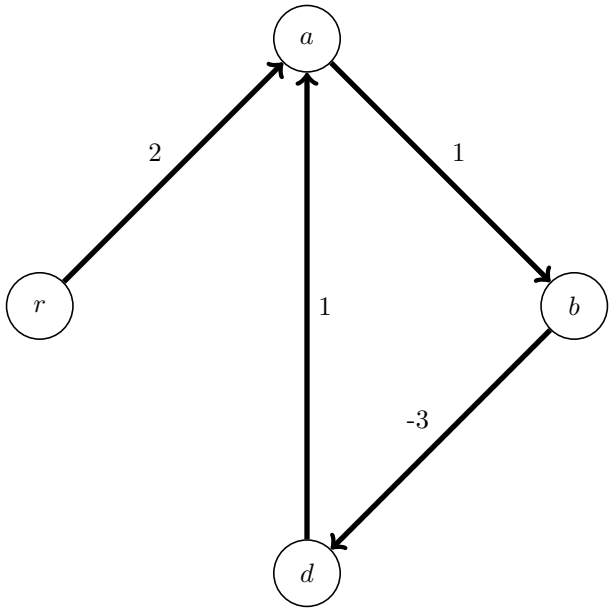


Vertices	Inicio		I_1 $vu = ra$		I_2 $vu = rb$		I_3 $vu = ad$		I_4 $vu = ba$		I_5 $vu = ad$	
-	y	p	y	p	y	p	y	p	y	p	y	p
r	0	0	0	0	0	0	0	0	0	0	0	0
a	∞	-1	3	r	3	r	3	r	2	b	2	b
b	∞	-1	∞	-1	1	r	1	r	1	r	1	r
d	∞	-1	∞	-1	∞	-1	5	a	5	a	4	a

Inicio: $y_v = \infty \ \forall v \in V - \{r\}$, $y_r = 0$
1ra iteración: Toma arco ra y actualiza: $y_a = 3$, ya que $y_r + w(ra) < y_a = \infty$. $p_a = r$
2da iteración: Elige arco rb y actualiza: $y_b = 1$, $p_b = r$.
3ra iteración: Toma arco ad y actualiza: $y_d = 5$, $p_d = a$.
4ta iteración: Toma arco ba y actualiza $y_a = 2$, porque $y_b + w(ba) < 3$. $p_a = b$
5ta iteración: Por último, toma ad y actualiza $y_d = 4$ porque $y_a + w(ad) < 5$. $p_d = a$
Grafo con los y_v en cada nodo despues de aplicado el algoritmo de Ford:



Ejemplo 2:



Vértices	Inicio		I_1 $vu = ra$		I_2 $vu = ab$		I_3 $vu = bd$		I_4 $vu = da$		I_5 $vu = ab$	
-	y	p	y	p	y	p	y	p	y	p	y	p
r	0	0	0	0	0	0	0	0	0	0	0	0
a	∞	-1	2	r	2	r	2	r	1	d	1	d
b	∞	-1	∞	-1	3	a	3	a	3	a	2	a
d	∞	-1	∞	-1	∞	-1	0	b	0	b	0	b

Inicio: $y_v = \infty \ \forall v \in V - \{r\}$, $y_r = 0$
1ra iteración: Toma arco ra y actualiza: $y_r + w(ra) < y_a = \infty \longrightarrow y_a = 2$ $p_a = r$
2da iteración: Toma arco ab: $y_b = 3$, $p_b = a$.
3ra iteración: Elije arco bd y actualiza: $y_d = 0$, $p_d = b$.
4ta iteración: Toma arco da y actualiza $y_a = 1$, porque $y_d + w(da) < y_a = 2$. $p_a = d$.
5ta iteración: Toma ab y actualiza $y_b = 2$, porque $y_a + w(ab) < 3$. $p_b = a$.
Si observamos el algoritmo seguirá indefinidamente dado que (a, b, d, a) es un ciclo negativo de valor $1 + 1 - 3 = -1$, entonces siempre conviene pasar por el ciclo.
Esto pasará siempre que tengamos un grafo con un circuito negativo.

1.3 Validez del algoritmo de Ford

Proposición 3: Si G no tiene circuitos dirigidos de costo negativo, entonces en cualquier iteración del algoritmo de Ford vale que:
(i) Si $y_v \neq \infty$, entonces es el costo de algún camino simple dirigido de r a v.
(ii) Si $p_v \neq -1$, entonces p_v es el predecesor de v en un camino simple de r a v, de costo a lo sumo y_v.

Teorema 4: Si G no tiene circuitos dirigidos de costo negativo, entonces el algoritmo de Ford termina luego de un número finito de iteraciones. Al terminar, p define un camino de costo mínimo de r a v de costo y_v para todo $v \in V$.
Demostración: Notemos que hay finitos caminos simples en G. Así, por la Proposición 3, hay un número finito de valores para y_v. Como en cada iteración del algoritmo y_v solo decrementa, el algoritmo termina. Al terminar, p_v define un camino simple de r a v de costo $\leq y_v$. Pero al ser y un potencial factible, por Proposición 2 el costo del camino no puede ser menor a y_v, entonces debe ser igual.
Teorema 5: Un grafo G tiene un potencial factible sí y solo sí no tiene circuitos dirigidos de costo negativo.
Proposición 6: Sea G un digrafo. Si w define costos de arcos enteros, y G no tiene circuitos de costo negativo, entonces el algoritmo de Ford termina luego de a lo sumo $C \times n^2$ operaciones, donde $C = 2 \times max(|w(e)| : e \in E)$
Podemos ver entonces que este algoritmo no es eficiente.
Teorema 7: Sea G es un digrafo con vector de costos reales w_e. Sean r,$w \in V$. Si existe un camino de costo mínimo de r a v para todo $v \in V$, entonces:

$$min\{w(P) : P \text{ es un camino dirigido de } r \text{ a } w\} = \max\{y_w : y \text{ es un potencial factible}\}$$

1.4 Mejoras al algoritmo de Ford

Si guardamos de forma correcta los valores de y, podemos realizar en tiempo constante cada iteración del algoritmo de Ford. Pero el número de iteraciones depende del orden en el que elegimos los arcos, pudiendo resultar esta elección en una ejecución muy lenta del algoritmo.
Definición 8: Sea G un digrafo con costos, $S = f_1, f_2, ..., f_l$ una secuencia de arcos de G, y P un r, v-camino. Decimos que P está embebido en S si sus arcos ocurren como una subsecuencia de S.
Proposición 9: Si el algoritmo de Ford utiliza la secuencia S, entonces para cada $v \in V$ y para cada r, v-camino P embebido en S, tenemos que $y_v \leq w(P)$.
Observación: Si S cumple que para todo $v \in V$ hay un camino de costo mínimo de r a v embebido en S, entonces el algoritmo de Ford termina si utiliza la secuencia S. Podemos utilizar esto para mejorar el algoritmo de Ford.

1.5 Algoritmo de Ford-Bellman

Podemos observar que si tenemos un digrafo G con n vértices, si tomamos la secuencia
$S = S_1, S_2, ..., S_{n-1}$, donde S_i es un ordenamiento (en principio, cualquiera) de los arcos de G, cualquier camino simple está embebido en S. Entonces por la observación anterior, si G no tiene circuitos dirigidos negativos, el algoritmo de Ford termina si utiliza la secuencia S. Además, termina en $m \times (n-1)$ operaciones.
Notemos que también podemos identificar si G tiene circuitos negativos: si luego de ejecutar el algoritmo de Ford con la secuencia S el vector y no es un potencial factible, significa que G tiene un circuito negativo. De lo contrario, el algoritmo ya hubiese terminado. De aquí surge el algoritmo de Ford-Bellman:

Entrada: Un grafo dirigido $G = (V, E)$, un vértice $r \in V$, y una función de costos $w(e) : e \in E$.

Inicio: Un vector y, donde $y_r = 0$, $y_v = \infty$ si $v \neq r$. Un vector p donde $p_r = 0$, $p_v = -1$ si $v \neq r$, que nos servirá para reconstruir el camino de r a v. Una secuencia $S = S_1, S_2, ..., S_{n-1}$, donde S_i es un ordenamiento de los arcos de G.

Iteración: Mientras $i < n$ e y no sea un potencial factible:

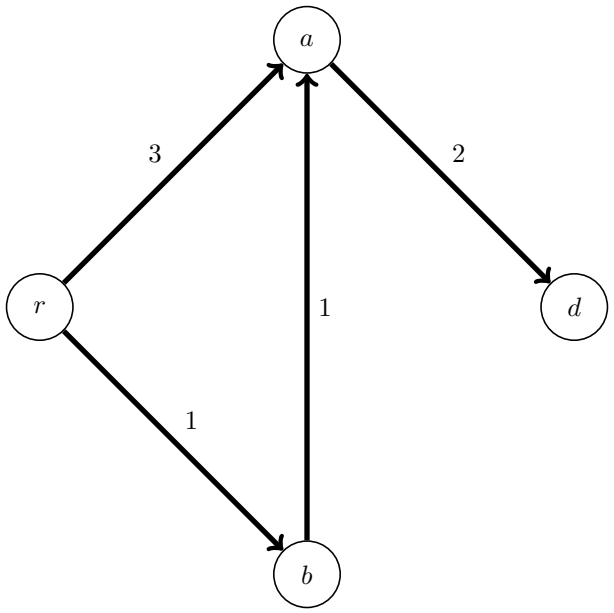
$i = i + 1$

Para cada arista $vu \in S_i$:

si $y_v + w(vu) < y_u$, corregir $y_u = y_v + w(vu)$.

Salida: Un potencial factible y con los costos mínimos de r a todos los vértices de V y un vector p donde p_v es el predecesor de v en el camino de menor costo de r a v. O que tiene un ciclo negativo y no se puede calcular y.

Teorema 10: El algoritmo de Ford-Bellman computa correctamente un camino dirigido de costo mínimo de r a v para todo $v \in V$ (si $i < n$ al terminar), o detecta que existe un circuito de costo negativo (si $i = n$ al terminar). En cualquiera de los casos corre en complejidad $O(mn)$.
Veamos los ejemplos anteriores ejecutando Ford-Bellman:
Ejemplo 3:



Al iniciar el algoritmo tenemos:

Vértices	Inicio	
-	y	p
r	0	0
a	∞	-1
b	∞	-1
d	∞	-1

y S = $\{S_1, S_2, S_3\}$ con
$S_1 = \{ra, rb, ba, ad\}$
$S_2 = \{ra, ba, ad, rb\}$
$S_3 = \{ad, ra, rb, ba\}$

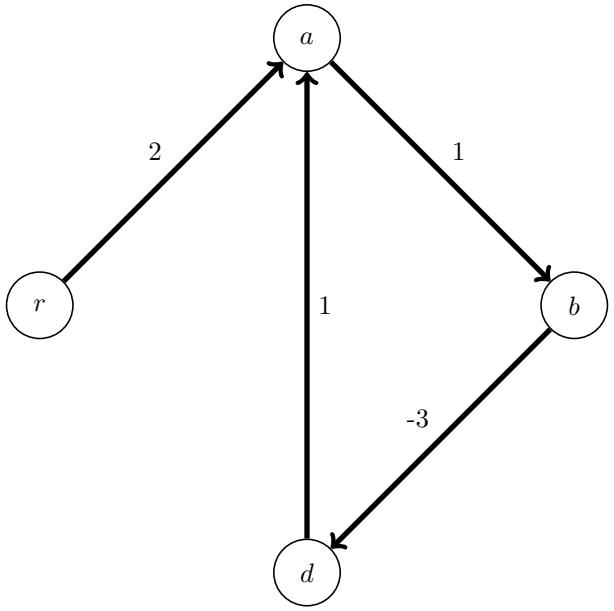
Primera iteración (S1):

Vértices	ra		rb		ba		ad	
-	y	p	y	p	y	p	y	p
r	0	0	0	0	0	0	0	0
a	3	r	3	r	2	b	2	b
b	∞	-1	1	r	1	r	1	r
d	∞	-1	∞	-1	∞	-1	4	a

ra: $y_r + w(ra) = 3 < \infty = y_a \longrightarrow y_a = 3$
rb: $y_r + w(rb) = 1 < \infty = y_b \longrightarrow y_b = 1$
ba: $y_b + w(ba) = 2 < 3 = y_a \longrightarrow y_a = 2$
ad: $y_a + w(ad) = 4 < \infty = y_d \longrightarrow y_d = 4$

Verificación condición y potencial factible:
ra: $y_r + w(ra) = 3 > 2 = y_a$ ✓
rb: $y_r + w(rb) = 1 = y_b$ ✓
ba: $y_b + w(ba) = 2 = y_a$ ✓
ad: $y_a + w(ad) = 4 = y_d$ ✓
Además $y_r = 0$. Podemos decir que y es potencial factible. Por lo tanto concluimos que el vector contiene todos los costos de los caminos más cortos desde r a cualquier vértice del grafo.

Ejemplo 4:



Al iniciar el algoritmo tenemos:

Vértices	Inicio	
-	y	p
r	0	0
a	∞	-1
b	∞	-1
d	∞	-1

y S = $\{S_1, S_2, S_3\}$ con
$S_1 = \{ra, da, ab, bd\}$
$S_2 = \{ra, ab, bd, da\}$
$S_3 = \{ad, bd, da, ab\}$
Primera iteración (S_1, i = 1):

Vértices	ra		da		ab		bd	
-	y	p	y	p	y	p	y	p
r	0	0	0	0	0	0	0	0
a	2	r	2	r	2	r	2	r
b	∞	-1	∞	-1	3	a	3	a
d	∞	-1	∞	-1	∞	-1	0	b

ra: $y_r + w(ra) = 2 < \infty = y_a \longrightarrow y_a = 2$
da: $y_d + w(da) = \infty > 2 = y_a \longrightarrow y_a = y_a$
ab: $y_a + w(ab) = 3 < \infty = y_b \longrightarrow y_b = 3$
bd: $y_b + w(bd) = 0 < \infty = y_d \longrightarrow y_d = 0$

Segunda iteración (S_2, i = 2):

Vértices	ra		ab		bd		da	
-	y	p	y	p	y	p	y	p
r	0	0	0	0	0	0	0	0
a	2	r	2	r	2	r	1	d
b	3	a	3	a	3	a	3	a
d	0	b	0	b	0	b	0	b

ra: $y_r + w(ra) = 2 = y_a \longrightarrow y_a = y_a$
ab: $y_a + w(ab) = 3 = y_b \longrightarrow y_b = y_b$
bd: $y_b + w(bd) = 0 = y_d \longrightarrow y_d = y_d$
da: $y_d + w(da) = 1 < 2 = y_a \longrightarrow y_a = 1$

Tercera iteración (S_3, i = 3):

Vertices	ra		bd		da		ab	
-	y	p	y	p	y	p	y	p
r	0	0	0	0	0	0	0	0
a	1	d	1	d	1	d	1	d
b	3	a	3	a	3	a	2	a
d	0	b	0	b	0	b	0	b

ra: $y_r + w(ra) = 2 > y_a \longrightarrow y_a = y_a$
bd: $y_b + w(bd) = 0 = y_d \longrightarrow y_d = y_d$
da: $y_d + w(da) = 1 = y_a \longrightarrow y_a = y_a$
ab: $y_a + w(ab) = 2 < 3 = y_b \longrightarrow y_b = 2$

Verificación condición y potencial factible:
ra: $y_r + w(ra) = 2 = y_a$ ✓
da: $y_d + w(da) = 1 = y_a$ ✓
ab: $y_a + w(ab) = 2 = y_b$ ✓
bd: $y_b + w(bd) = -1 < 0 = y_d$ X

Como $i = 4$ e y no es potencial factible ya que existe la arista bd tal que $y_b + w(bd) = 2 - 3 = -1 < 0 = y_d$ concluimos que el grafo tiene un ciclo negativo y no es posible hallar el camino mas corto a todos los vértices desde r.
Podemos ver una diferencia en este ejemplo, Ford-Bellman puede identificar el ciclo negativo, y no itera infinitamente como Ford.